Using AutoEncoders for Radio Signal Denoising

Ebtesam Almazrouei
Emirates ICT Innovation Center (EBTIC), Khalifa University of Science and Technology (KUST)
Abu Dhabi, UAE
ebtesam.almazrouei@ku.ac.ae

Gabriele Gianini
EBTIC, Khalifa University (KUST)
Abu Dhabi, UAE
Università degli Studi di Milano
Milano, Italy
gabriele.gianini@unimi.it

Corrado Mio
Dipartimento di Informatica "Giovanni degli Antoni" Università degli Studi di Milano
Milano, Italy
corrado.mio@unimi.it

Nawaf Almoosa
Emirates ICT Innovation Center (EBTIC), Khalifa University of Science and Technology (KUST)
Abu Dhabi, UAE
nawaf.almoosa@ku.ac.ae

Ernesto Damiani
EBTIC, Khalifa University (KUST)
Abu Dhabi, UAE
Università degli Studi di Milano
Milano, Italy
ernesto.damiani@unimi.it

ABSTRACT

We investigated the use of a Deep Learning approach to radio signal de-noising. This data-driven approach has does not require explicit use of expert knowledge to set up the parameters of the denoising procedure and grants great flexibility across many channel conditions. The core component used in this work is a Convolutional De-noising AutoEncoder, known to be very effective in image processing. The key of our approach consists in transforming the radio signal into a representation suitable to the CDAE: we transform the time-domain signal into a 2D signal using the Short Time Fourier Transform. We report about the performance of the approach in preamble denoising across protocols of the IEEE 802.11 family, studied using simulation data. This approach could be used within a machine learning pipeline: the denoised data can be fed to a protocol classifier. A perspective advantage of using the AutoEncoders in that pipeline is that they can be co-trained with the downstream classifier, to optimize the classification accuracy.

CCS CONCEPTS

• **Computing methodologies → Neural networks**.

KEYWORDS

Deep Learning, AutoEncoders, Signal Denoising, Radio Spectrum

ACM Reference Format:
Ebtesam Almazrouei, Gabriele Gianini, Corrado Mio, Nawaf Almoosa, and Ernesto Damiani. 2019. Using AutoEncoders for Radio Signal Denoising. In *15th ACM Symposium on QoS and Security for Wireless and Mobile Networks (Q2SWinet'19), Nov. 25–29, 2019, Miami Beach, FL, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3345837.3355949

Q2SWinet '19, November 25–29, 2019, Miami Beach, FL, USA

ACM ISBN 978-1-4503-6906-0/19/11…$15.00
https://doi.org/10.1145/3345837.3355949

1 INTRODUCTION

The adoption of intelligent techniques in the management of spectrum sharing can support the coexistence of heterogeneous radio-access technologies and the significantly improve capacity and spectrum utilization. However, it has to face some key challenges: the effectiveness of the algorithms is required to generalize across radio-access scenarios; furthermore, the adopted solutions must be easily applicable across multiple radio standards.

Machine learning (ML) methods, and, more specifically, a set of recently developed techniques, known as Deep Learning (DL) [8], bear the potential of advancing the intelligence of radio devices, providing data-driven flexible solutions, without relying heavily on expert knowledge. Among the problems that the ML can target are protocol detection, and classification, and signal denoising; further applications might include device or user profiling and classification, source counting.

Here we focus on the problem of signal denoising: we apply a Deep Learning model to protocol preambles. By noise, here we mean the ensemble of channel effects, which degrades the signal up to reception. Specifically, we propose a method to unfold the *spectrogram* of the transmitted preamble from noise by means of a Convolutional Denoising AutoEncoder (CDAE, defined in Section 2). Since such a spectrogram is essentially a 2D image, the convolutional approach is very effective. We illustrate the technique using simulated data of protocols: IEEE802.11n and IEEE 802.11a.

After denoising the signal, one could in principle apply to it a classifier (e.g. a standard or a convolutional multi-layer perceptron (MLP)). An advantage of using CDAEs is that – differently from other traditional techniques used for denoising – they can be co-trained with the downstream classifier, to optimize accuracy. Obviously, CDAEs and classifiers can also be trained separately if the denoising already achieves satisfactory results, as it happens with our application case: we have a reconstruction loss of the order around 1% – 5%, depending on the SNR. Despite these results, we stress that this work does not aim at competing with traditional well-established denoising techniques in terms of accuracy: the point is that most of those techniques are based on domain expert knowledge, whereas our method is completely data-driven: thus it allows for high flexibility.

Notice that the CDAEs could be used also within a classification pipeline, to enhance the classification performance. Their advantage would be that not only they can be trained in isolation, but also co-trained with the classifier algorithm, so as to improve the overall pipeline performance: this option would not be available using more traditional denoising techniques.

The present paper is structured as follows: Section II recalls the definitions of ML, DL, and the related work of ML applications to radio signal processing. In section III, Denoising AutoEncoders are described. Data sets and analysis tools are discussed in Section IV. The architecture of the proposed Denoising AutoEncoders, loss function, and optimization algorithms are discussed in Section V. The experimental configurations and results are discussed in Section VI; conclusions are drawn in Section VII.

2 DEFINITIONS AND RELATED WORK

2.1 Deep Learning: Convolutional AutoEncoders

A special role in the recent developments of Machine Learning has been taken by Artificial Neural Network algorithms (ANNs). The most common ANN is the Multi-Layer Perceptron (MLP) used both for classification and regression. An MLP learns from examples a certain set of input-output mappings, by optimizing the weights that link successive layers of artificial neurons [16]. In the latest few years, new architectures and training methods have been developed for ANNs: they go under the name of Deep Learning (DL) methods. DL methods allow the use of larger and more complex models and can face harder tasks, using reasonable computational time and hardware resources.

Among the DL architectures are the Convolutional Neural Networks (CNNs) and the AutoEncoders (AEs): those two techniques are used together to create Convolutional AutoEncoders (CNN-AE). Hereafter, we explain briefly their approach to automatic learning. Differently from standard fully connected MLP, where every inner neuron can feed all the neurons of the next layer, in CNNs there is a limited number of connections from one layer to another, and the same pattern of weights is used for a layer to the next; this, and the presence of special down-sampling layers (the "pooling" layers) greatly reduces the number of parameters search space. CNNs are very effective in image processing since they exploit image coherence and local correlation.

AutoEncoders (AEs), are multilayered ANNs characterized by a symmetric input-encoding-output architecture and a special training procedure. AEs are often used to find a compact data *encoding*. They learn by self-supervision, i.e. they do not try to output a label associated to an input example, but try to replicate the input example on the output, i.e. use each input record as its own (structured) label. The standard architecture of an AE consists of at least three layers: an input layer, a hidden layer, and an output layer. The hidden layer (the encoding layer) is (in dimensionality reduction) smaller than the input. AEs are trained using the back-propagation algorithm, then their learned weights are frozen, and the layer(s) beyond the central (encoding) one are discarded. What is left is a function that maps the input into a different representation, which presented to a classifier learning algorithm typically yields performance improvements. Stacking successively trained AEs one can build very deeper models, by optimizing just a moderate number of parameters at a time. AEs can be used also independently from a downstream classifier: indeed finding a better (in some sense) representation is a worthwhile result in itself. Denoising AEs (see below) are trained by providing them a noisy input and challenging them to output, not the input itself, but a non-noisy version of it (see Figure 1): they try to find a representation where the signal is unfolded from noise.

CNN-AEs merge the advantages of the AutoEncoders representation learning and the reduced complexity of the CNN paradigm. When trained for finding a denoised representation the CNN-AEs are called Convolutional De-noising AutoEncoders (CDAEs).

2.2 Signal Denoising by ML Techniques

Several ML models have been applied to radio signals problems. Here we restrict to denoising applications.

In [7], H-SVMs have been proposed to estimate the noise level for an Additive White Gaussian Noise (AWGN) channel in MIMO wireless network. In [14], the CNN classifier is proposed for modulation classification in presence of different Signal-to-Noise Ratios (SNRs). In [15], CNNs are used for spectral identification over different SNRs. In this paper, we apply the CNN AEs (CDAE) to denoise the radio signals from the fading channel effects. We are not aware of attempts to improve the signal quality using ML models *before* signal identification, under fading channel conditions.

3 DE-NOISING AUTOENCODERS

Here we briefly describe the Denoising AutoEncoders [18], which are used to reconstruct data from a corrupted input. In contrast to standard AEs – which receive in input an example and is trained to reconstruct the input as faithfully as possible – the Denoising AE are given in input a noisy example and forced to reconstruct a denoised version. To do so, one needs both the non-noisy and the noisy version of each example: those can be obtained, for instance, from a radio simulation environment (input=Rx, target output=Tx), or by corrupting artificially non-noisy data, or by measuring the Tx and Rx signals in a physical environment.

AutoEncoders [18] consists of an encoder which represents the mapping F_Θ of the input vector X into hidden representation $y = f_\Theta(x) = s(Wx + b)$ where s is a nonlinear function such as Rectified Linear Unit (ReLU) or a sigmoid, while Θ represents the set of parameters $\Theta = (W, b)$. W is the weight matrix and b is the offset vector. The representation y is then decoded into an array z (with the same size as the input x), by $z = g_\Theta(y) = s(W'y + b')$. The parameters W, W', b, and b' are optimized by minimizing a suitable loss function. For binary data where $x \in [0,1]^d$, the loss function typically used is Mutual Information (MI) [17]: $MI \equiv -\sum_j [x_j log(z_j) + (1 - x_j) log(1 - z_j)]$.

De-noising AutoEncoders (DAEs) follow the general concept of AutoEncoder but use a noisy input $\hat{x} \neq x$. The reconstructed z is however compared to the clear signal x in the loss function.

Such a loss function is very often minimized by means of the stochastic gradient descent (SGD) algorithm [10]. We chose for our model a variant of SGD, the Adaptive Gradient Descent (AGD) [6].

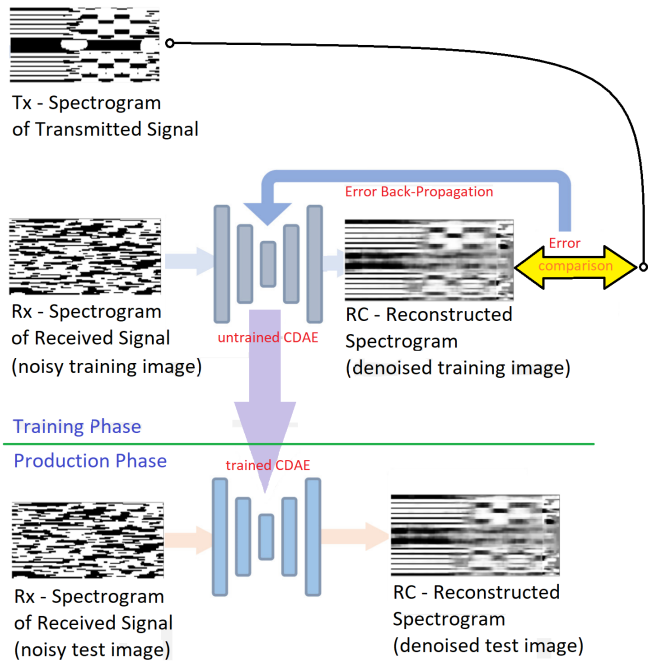

Figure 1: Schematic view of the operation of a Convolutional De-noising Auto-Encoder (CDAE). The input can consist in both noisy and non-noisy data in some proportion: here the noisy data are the spectrograms of the received signal, (generated propagating the transmitted signal in the non-ideal channel). The input spectrograms are processed by the encoder, which consists of multiple convolutional layers, and by the decoder: the output is compared to the spectrogram of the clean signal. By optimizing the loss function, the network finds a compressed representation which is to some extent denoised.

The main difference between the two is that in the latter, the learning rate is not fixed, but depends on the loss value. We preferred AGD because in general it is known to work better for images.

In our case, we used convolutional layers (i.e. we used a Convolutional De-noising AutoEncoder): the parameters W and b of a patch of the image are shared among all the locations to enforce spatial locality. In general, CDAEs perform better than classical DAEs in image processing [10].

4 SIMULATION DATA

We study the simplified setting in which we know that a single source is transmitting, this assumption will be lifted in future works. We performed several experiments related to distinct multi-path fading channel conditions (presence or absence of Doppler effect, different Signal-to-Noise Ratios), with the two protocols IEEE 802.11a, and IEEE 802.11n. We studied the performances of CDAE at the following tasks:

- de-noising of received signals from one known protocol (either IEEE 802.11a or IEEE 802.11n);
- de-noising of the received signals from an unknown protocol taken from a set of two protocols (the received protocol was either IEEE 802.11a or IEEE 802.11n).

We carried on the investigation using of simulated data, and focusing on the reconstruction of the *preamble* of each protocol. For the sake of simplicity, we implicitly assumed that the denoiser was provided in input with a well-delimited preamble. Of course, in practice, this would imply a preliminary phase in which the preamble is singled out from the received data stream. This task can be accomplished with a segmentation algorithm, however, its specification is out of the scope of the present paper. The operational issues from the problem will be discussed in a future work.

4.1 Simulation and analysis tools

We used mainly MATLAB for generating simulated data and the TensorFlow and Keras libraries (in Python) for building the Deep Learning models.

Specifically, we used the MATLAB/SIMULINK suite to simulate the whole model of the physical layer of the IEEE wireless local area network (WLAN). The WLAN System Toolbox allows the configuration of the physical layer waveform for each IEEE standard and makes it possible to design the transmitter, the channel model, and the receiver [11]. For example, we used the *wlanTGnChannel* component to pass input signals Tx through an 802.11n (TGn) multipath fading channel. Various parameters could be set up for a specific WLAN scenario: the sample rate fs, channel bandwidth cbw, large-scale fading effect, path loss and shadowing, delay profile, and the channel model.

Several tools have been used for building artificial neural networks. TensorFlow (TF) is a set of Python libraries to develop deep neural networks [2]. We used the Keras libraries to control the TF back-end [4].

The simulation and the learning procedures were carried on both under Windows 10 and Linux, utilizing graphical processing units (GPUs).

4.2 Noise model used in the data

We focussed on Wi-Fi signals operated in 2.4GHz and 5GHz range. Specifically, we generated the end-to-end radio signal environment for the following standards: IEEE 802.11a and IEEE 802.11n. The design parameters for each protocol (channel bandwidth BW, carrier frequency f_c, modulation scheme, channel model, number of transmitters and receivers, etc.) are summarized in Table 1 [1, 3, 9].

Table 1: WLAN Standards Characteristics

Protocol	Maximum Data rate	Channel Bandwidth	Frequency Band (GHz)	Modulation
IEEE 802.11n	600 Mbps 4 spatial streams	20MHz, 40MHz	2.4 GHz, 5 GHz	OFDM
IEEE 802.11a	54Mbps	20MHz	5GHz	OFDM

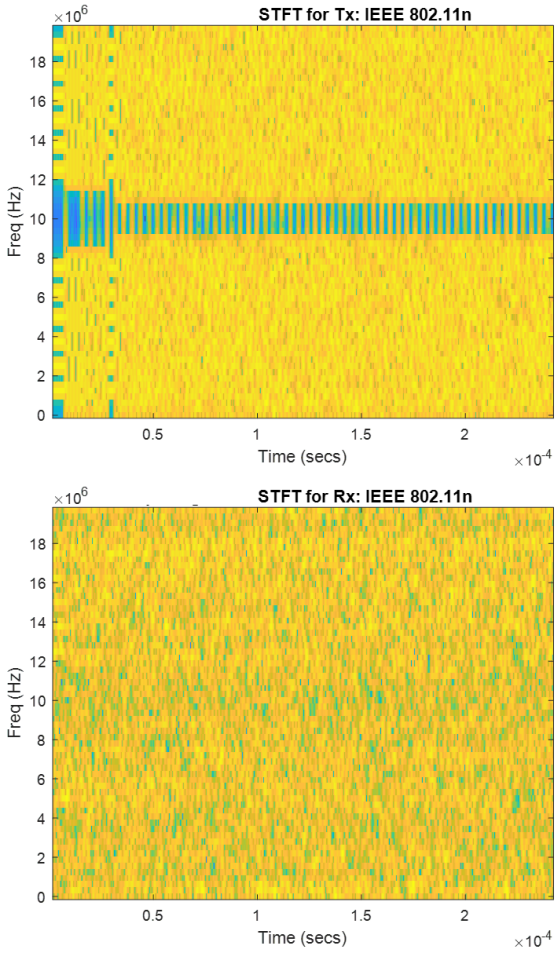

Figure 2: Spectrograms for radio signals for IEEE 802.11n. Top: transmitted signals (Tx). Bottom: received signals (Rx) under the hypothesis of SNR 0 dB and NLOS fading channel described in Table 2 (standard model E from reference [12]).

Table 2: Rayleigh channel model E

Path Delays (nanoseconds)	[0 10 20 40 70 100 140 190 240 320 430 560 710 880 1070 1280 1510 1760]
Average path gains (dB)	[-4.9 -5.1 -5.2 -0.8 -1.3 -1.9 -0.3 -1.2 -2.1 0.0 -1.9 -2.8 -5.4 -7.3 -10.6 -13.4 -17.4 -20.9]

In the practical radio spectrograms, the radio signal suffers from noise due to the No-Line of Sight (NLOS), multi-path, inter-symbol interference, Doppler effect, and fading effects. In our experiments, the radio signals were modeled by a Rayleigh channel and NLOS effects. Several standard indoor channel models have been developed in [12]. In our research, we chose to study model E, corresponding to a typical large open space, indoor or outdoor, with large delay spread (we set 250 ns delay spread), a highly dispersive channel. Table 2 details path delays and average path gains for Model E.

4.3 The datasets

Our dataset consisted of 1000 radio spectrogram images corresponding to as many preambles for each studied IEEE 802.11 protocol. The specific details of the experimental configurations, for which the quantized spectrograms have been generated, are detailed below, in Section 6.

4.3.1 Spectrograms. Each spectrogram represented the Short Time Fourier Transform (STFT) [13] of the raw time series $x(t)$ corresponding to the signal of a preamble. More specifically, a spectrogram pictures the STFT $S_x(\tau, \omega)$ as a function of the (discretized) time τ and frequency ω. Each preamble was partitioned into 3782 time intervals, for each interval 64 frequencies were computed. Thus, each image consisted in 64×3782 gray-level pixels, where gray-level expressed the modulus $|S_x(\tau, \omega)|$.

Examples of spectrograms are shown in Figure 2 where the different levels of intensity are mapped into different colors. One can notice the difference between the transmitted signal spectrogram and the received signal spectrograms, i.e. the degradation undergone by both protocol preambles, due to the channel.

4.3.2 Quantization. The spectrogram has very high dimensionality. To reduce it, the values $|S_x(\tau, \omega)|$ have been quantized and encoded with binary values $b_x(\tau, \omega)$ as follows: $b_x(\tau, \omega) = 1$ if $|S_x| > \langle S_x(\tau, \omega) \rangle$ (where the average $\langle \cdot \rangle$ is taken over the spectrogram pixels) and zero otherwise.

4.4 Trimming and padding

One can also observe that there are regions of the transmitted signal that seem not to be carrying special visual information. We exploit this observation in order to simplify the reconstruction task: we trim away those uninformative regions. Specifically, the regions corresponding to the upper third and the lower third of the left side of each spectrogram have been padded with zeros and excluded from the learning and reconstruction process.

Dropping those pixels, i.e. dropping those features, is expected to be uninfluential on the parts that a hypothetical classifier (downstream of the denoiser) would use.

Examples of quantized and trimmed spectrograms are shown in Figure 3 which illustrates the process.

5 DENOISING WITH CDAES

The problem we address hereafter is the denoising of the preamble spectrograms of a single radio protocol at a time. In other words, we adopt the simplifying assumptions that, in the sensed area, only one protocol is used and only one transmitter is operational. Furthermore, we posit that we are able to identify the time interval in which lies the preamble and distinguish it from the subsequent part of the frame. To de-noise the radio spectrograms, we used the CDAEs introduced in Section 3.

Already from the example of spectrograms shown in Figure 1, one can see that such a 2D signal presents regions of qualitatively homogeneous behavior: e.g. one can notice a horizontal central strip, consisting in vertical strips of low values, alternated to higher values.

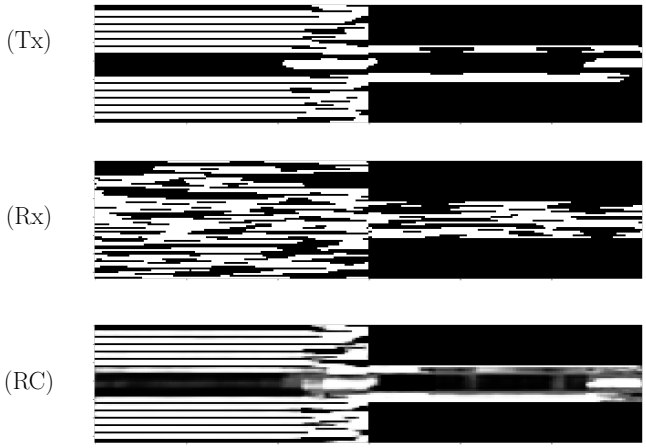

Figure 3: Quantized spectrograms of preambles for IEEE 802.11n: transmitted spectrogram (Tx), received spectrogram (Rx) (with 0 dB SNR) and the spectrogram reconstructed (RC) using the CDAE neural network. The black regions on the left represent parts of the spectrogram not used in the learning process, on account of their apparent lack of distinctive visual content (compare with Figure 2).

The spectrogram can be considered a sort of picture, with coherent regions. The reasons why we chose to experiment with convolutional versions of the AE are several, however we counted on the fact that the convolutional mechanisms are known to be effective in processing/learning images.

It is true that the special kind of image under study had rather unbalanced dimensions (64×3782 pixels per spectrogram instance), but the internal coherence of different regions of the image suggested that the spectrogram could be dealt with almost like an ordinary gray-scale image.

Among the main parameters to choose for setting up a CDAE are the loss function and the optimization algorithm for the neurons' link weights. As an evaluation metric for CDAE several loss functions can be used: we used binary cross entropy.

The CDAE follows the standard approach of de-noising AutoEncoders with encoded and decoded convolution neural layers but uses also some convolutional layers. Convolutional layers are known to preserve the spatial image distribution so that the corresponding networks perform better than non-convolutional neural networks in image processing [10].

The operation of a CDAE network was shown in Fig. 1.

The architecture of CDAE we used is summarized in Table 3. It encompasses a sequence of sixteen layers: an input layer, seven 2D convolutional layers for the *encoder*, and eight 2D convolutional layers for *decoder*. The 2D *encoder* includes convolutional layers, use *rectified linear unit (ReLU)* layers, a *dropout* layer, and a 2D *max-pooling* layer. The *ReLU* layer computes element-wise the maximum between a value and a threshold. The *dropout* layer is used for better generalization and to avoid overfitting. The *max-pooling* performs a deterministic down-sampling operation to reduce the spatial dimensions (width, height) of the convolutional layer.

The first hidden layer is a convolutional layer with 16 feature maps. Each feature map is connected to a different kernel. Each kernel has size 3 by 3 in each feature map. The *decoder* convolutional layers includes a *ReLU* layer, an *up-sampling* layer, and a *dropout* layer. The *up-sampling* layer is adapted to change the output of convolutional layers in the decoder to a higher resolution that matches the original input size image. The AE is trained at the learning rate 0.001 [5].

Table 3: CDAE model architecture

Layer Type	Output Shape
Input layer	(None, 300, 64, 1)
2D convolutional layer (Conv2D)	(None, 300, 64, 16)
Dropout layer	(None, 300, 64, 16)
MaxPooling	(None, 150, 32, 16)
Conv2D	(None, 150, 32, 16)
MaxPooling	(None, 75, 16, 16)
Conv2D	(None, 75, 16, 16)
MaxPooling	(None, 25, 8, 16)
Conv2D	(None, 25, 8, 16)
Upsampling	(None, 75, 16, 16)
Conv2D	(None, 75, 16, 16)
Upsampling	(None, 150, 32, 16)
Conv2D	(None, 150, 32, 16)
Dropout	(None, 150, 32, 16)
UpSampling	(None, 300, 64, 16)
Conv2D	(None, 300, 64, 1)

6 EXPERIMENTAL CONFIGURATIONS AND RESULTS

6.1 Studied Configurations

The WLAN environment for both IEEE 802.11a and IEEE 802.11n has been modeled as Single Input Single Output (SISO) consists of one transmitter Tx, a channel model, and one receiver Rx. The specification parameters for IEEE standards as mentioned in section 4.2 were followed. The channel has been modeled as a Rayleigh channel. Different levels of Signal-to-Noise-Ratio (SNR) were also introduced to study the robustness of the neural network performance. The simulation model included also the Doppler Effect.

For all transmitted and received packets for both IEEE standards, the signal was mapped into the corresponding spectrogram.

6.2 Results and Discussion

Overall, we generated 40,000 spectrogram images, a half for IEEE 802.11a and a half for IEEE 802.11n): 90% was used for the training and 10% for the test. Half of the training spectrograms were non-noisy, half were noisy. Table 4 summarizes these data.

Fig. 3 shows a sample results of the denoising process on the trimmed and padded spectrograms. It is apparent that the developed CDAE is capable of performing an effective denoising operation for the IEEE protocol preambles considered. This is due to the application of convolutional layers as they preserve the spatial locality of the input image.

Table 4: Size of the datasets in terms of number of images

Data Type	Noisy Datasets	Clean Datasets	Testing Datasets	Training Datasets
IEEE 802.11a	10000	10000	2000	18000
IEEE 802.11n	10000	10000	2000	18000
IEEE 802.11a and IEEE 802.11n	20000	20000	4000	36000

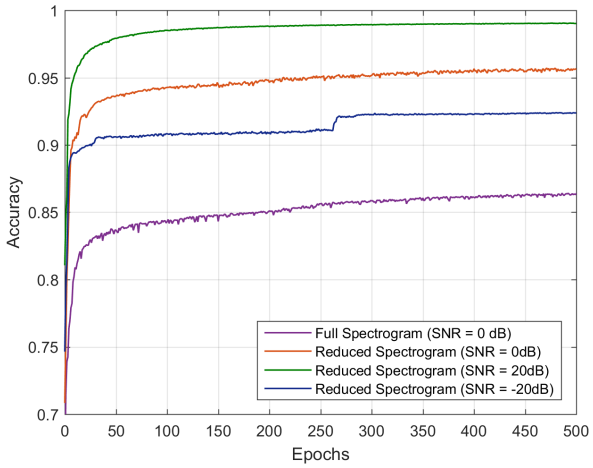

Figure 4: Test reconstruction accuracy of the developed CDAE model as a function of the number of training epochs for the full spectrogram (SNR=0 dB) and for the Reduced Spectrogram under different SNR conditions (SNR=0 dB, SNR=+20 dB, SNR=−20 dB).

The reconstruction accuracy of signal de-noising has been assessed with different metrics. Figure 4 shows the behavior of the *testing accuracy* defined in terms of *cross-correlation*, as a function of the number of training epochs. Figure 5 shows the behavior of the *testing loss* defined in terms of *average squared difference*. We consider three different SNR conditions: SNR=−20, 0, 20 dB.

Referring to Figure 4, for the case SNR=0 dB (i.e. power of the signal equal to the power of the noise), the accuracy of denoising the trimmed/reduced spectrograms reached the 95% after about 500 epochs (we ran the learning process up to 2000 epochs slowly increasing the accuracy up to 96%, but, for the sake of clarity, those results are not shown here). The training was stopped at 2000 epochs because further gains after that point were negligible. The reconstruction accuracy was approximately 4% lower for $SNR = -20$ dB, and 4% higher for $SNR = 20$ dB. For comparison, show also the untrimmed spectrogram outcomes (86% accuracy).

In Figure 5, one can see the box-plots of the errors for three different conditions SNR=−20, 0, 20 dB, plus the full spectrogram at SNR=0 dB: one can see that the latter has a higher loss with respect to the trimmed case at SNR=0 dB.

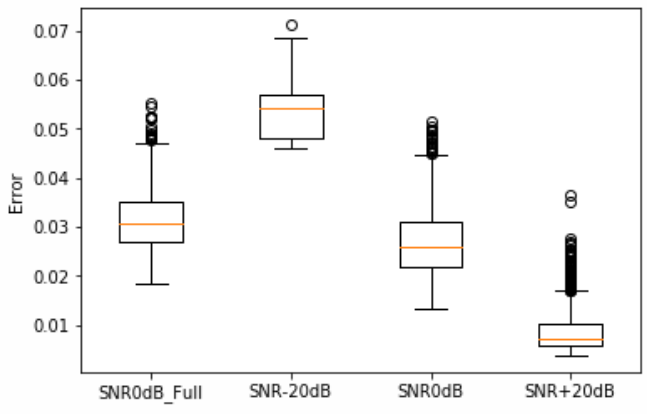

Figure 5: The distribution of the loss value for different SNR value. Each box plot corresponds to a different SNR condition. From left to right: Full spectrogram (no trimming) and SNR=0 dB; Reduced Spectrogram SNR=−20 dB; SNR=0 dB; SNR=+20 dB. The results were obtained using a model trained for 2000 training epochs: each boxplot summarizes 2000 test examples. The loss of an example is defined as the squared distance averaged over the pixels.

7 CONCLUSION

We proposed a system for signal de-noising based on machine learning algorithms and demonstrate its effectiveness using various WLAN protocols. The advantage of such systems is that they can perform such a task without relying on expert knowledge: this can potentially provide high flexibility. The de-noising algorithm for radio protocols was developed designing and training a Convolutional De-noising AutoEncoder (CDAE). The results show that the system is capable of performing de-noising for radio protocol and of reconstructing the transmitted radio signals even in presence of severe noise in the radio spectrograms. The overall accuracy of the training (1-loss) is about 95%.

Although we demonstrated the approach for protocols of the IEEE 802.11 family, this approach could be generalized to all radio protocols: further studies are required to examine the challenges and the effectiveness of such generalization.

The current study was performed using simulation data; we plan to apply this approach to data sets of real radio signals.

Among the assumptions of the current study was that only one radio source was active. Future investigations will consider multiple concurrent sources and the problem of filtering out the reciprocal interference.

We assumed that the denoiser was provided a well-delimited preamble: of course, this implies a preliminary phase in which the preamble is singled out from the received data stream. We will specify and study this part of the procedure, which can be mapped into a segmentation task, in a future work.

We plan, finally, to optimize the Convolutional AutoEncoder structure within an actual classification pipeline so as to improve the classification performance: this would be the main application scenario for this kind of denoising.

ACKNOWLEDGMENTS

The authors acknowledge the support of the ICT Fund at Khalifa University, Abu Dhabi (Project No. 88434000029). The work was partially funded by the EU H2020 Research Programme, within the projects Toreador (No. 688797) and Threat-Arrest (No. 786890).

REFERENCES

[1] [n.d.]. Wi-Fi: Overview of the 802.11 Physical Layer and Transmitter Measurements. http://download.tek.com/document/37W_29687_0_press.pdf. Accessed: 2018-09-09.

[2] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: a system for large-scale machine learning.. In *OSDI*, Vol. 16. 265–283.

[3] Sourangsu Banerji and Rahul Singha Chowdhury. 2013. On IEEE 802.11: Wireless LAN Technology. *arXiv preprint arXiv:1307.2661* (2013).

[4] François Chollet et al. 2015. Keras.

[5] Sebastian Dörner, Sebastian Cammerer, Jakob Hoydis, and Stephan ten Brink. 2018. Deep learning based communication over the air. *IEEE Journal of Selected Topics in Signal Processing* 12, 1 (2018), 132–143.

[6] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 12, Jul (2011), 2121–2159.

[7] Vin-sen Feng and Shih Yu Chang. 2012. Determination of wireless networks parameters through parallel hierarchical support vector machines. *IEEE Transactions on Parallel and Distributed Systems* 23, 3 (2012), 505–512.

[8] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep learning*. Vol. 1. MIT press Cambridge.

[9] Vaishali D Khairnar and Ketan Kotecha. 2013. Performance of vehicle-to-vehicle communication using IEEE 802.11 p in vehicular ad-hoc network environment. *arXiv preprint arXiv:1304.3357* (2013).

[10] Donghoon Lee, Sunghoon Choi, and Hee-Joung Kim. 2018. Performance evaluation of image denoising developed using convolutional denoising autoencoders in chest radiography. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 884 (2018), 97–104.

[11] WLAN Mathworks. 2017. System Toolbox.

[12] Jonas Medbo. 1998. Channel models for HIPERLAN/2 in different indoor scenarios. *ETSI/BRAN* (1998).

[13] Alfred Mertins and Dr Alfred Mertins. 1999. *Signal analysis: wavelets, filter banks, time-frequency transforms and applications*. John Wiley & Sons, Inc.

[14] Timothy J O'Shea, Johnathan Corgan, and T Charles Clancy. 2016. Convolutional radio modulation recognition networks. In *International conference on engineering applications of neural networks*. Springer, 213–226.

[15] Timothy J O'Shea, Tamoghna Roy, and Tugba Erpek. 2017. Spectral detection and localization of radio events with learned convolutional neural features. In *Signal Processing Conference (EUSIPCO), 2017 25th European*. IEEE, 331–335.

[16] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1985. *Learning internal representations by error propagation*. Technical Report. California Univ San Diego La Jolla Inst for Cognitive Science.

[17] John Shore and Rodney Johnson. 1980. Axiomatic derivation of the principle of maximum entropy and the principle of minimum cross-entropy. *IEEE Transactions on information theory* 26, 1 (1980), 26–37.

[18] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research* 11, Dec (2010), 3371–3408.